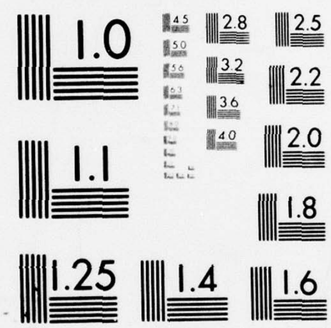
AD-A046 452 MARYLAND UNIV COLLEGE PARK COMPUTER SCIENCE CENTER F/G 17/2
RECURSIVE COMPUTATION OF PSEUDOINVERSE FOR APPLICATIONS IN IMAG--ETC(U)
SEP 77 E V KRISHNAMURTHY AFOSR-77-3271
UNCLASSIFIED TR-580 AFOSR-TR-77-1252 NL

| OF |
AD A046452

END
DATE
FILMED
12-77
DDC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

COMPUTER SCIENCE TECHNICAL REPORT SERIES

UNIVERSITY OF MARYLAND

COLLEGE PARK, MARYLAND
20742

(12)

(14) TR-580
(15) AFOSR-77-3271

(11) Sep[redacted]977

Approved for public releas
distribution unlimited.

(12) 13p.

(6) RECURSIVE COMPUTATION OF PSEUDOINVERSE FOR APPLICATIONS IN IMAGE RESTORATION AND FILTERING.

(10) E. V. Krishnamurthy
Computer Science Center
University of Maryland
College Park, MD 20742

(18) AFOSR

(19) TR-77-1252

(9) Technical rept.,

ABSTRACT

A bisection-transposition algorithm is described for the recursive computation of the Moore-Penrose inverse of a large matrix.

The support of the Directorate of Mathematical and Information Sciences, U. S. Air Force Office of Scientific Research, under Grant AFOSR-77-3271, is gratefully acknowledged, as is the help of Mrs. Shelly Rowe in preparing this paper. The continued interest and encouragement shown by Azriel Rosenfeld are deeply appreciated.

1473

403 018

LB

1. Definitions

Let A be a rectangular real (mxn) matrix, and X be a (nxm) real matrix satisfying the following conditions (t denotes transpose):

$$AXA = A \tag{1}$$

$$XAX = X \tag{2}$$

$$(AX)^t = AX \tag{3}$$

$$(XA)^t = XA \tag{4}$$

If X satisfies only (1) it is called a generalized inverse, and will be denoted by $A^{g_1} = X$. [1]; when X satisfies (1) and (2) we call $X = A^{g_2}$ a reflexive generalized inverse; when X satisfies (1), (2), (3) and (4) it is denoted by A^+ and is called the Moore-Penrose pseudoinverse.

The partitioning of a matrix A into submatrices A_1 and A_2 along the column (or row) direction will be denoted by

$$A = A_1 | A_2 \quad \left(\text{or } \frac{A_1}{A_2}\right).$$

2. Partitioning Methods

The computation of the pseudoinverse of a large matrix is very important for several applications in image processing and filtering [2]. Since the matrices involved in these problems are very large, it becomes necessary to use block partitioning schemes for this purpose. Unfortunately, however, the various methods used for block partitioning have several theoretical limitations, if one wants to set up a recursive scheme.

For instance, consider the conventional partitioning [1]

$$A = \begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array} \tag{5}$$

where A_{11}, A_{12}, A_{21}, A_{22} are arbitrary rectangular matrices.

Then

$$A^{g_2} = \left[\begin{array}{c|c} Q^{g_2} & -Q^{g_2} A_{12} A_{22}^{g_1} \\ \hline -A_{22}^{g_1} A_{21} Q^{g_2} & A_{22}^{g_2} + A_{22}^{g_1} A_{21} Q^{g_2} A_{12} A_{22}^{g_1} \end{array}\right] \tag{6}$$

$$A^{+} = \left[\begin{array}{c|c} Q^{+} & -Q^{+} A_{12} A_{22}^{g_1} \\ \hline -A_{22}^{g_1} A_{21} Q^{+} & A_{22}^{+} + A_{22}^{g_1} A_{21} Q^{+} A_{12} A_{22}^{g_1} \end{array}\right] \tag{7}$$

where $Q = A_{11} - A_{12} A_{22}^{g_1} A_{21}$. (8)

For these expressions to be valid, the following conditions are necessary:

(i) A should be positive semi-definite for using (6)

(ii) A should be positive semi-definite and Rank (A) = Rank (A_{11}) + Rank (A_{22}) for using (7).

Therefore, if we want to compute A^+ recursively, by a successive partitioning procedure involving smaller matrices, then at any stage the intermediate partitions or block matrices should also satisfy the above conditions. It is clear that while it is possible to make A positive definite by starting with AA^t and computing A^+ using the formula

$$A^+ = A^t(AA^t \cdot AA^t)^{g_2} AA^t$$

it cannot always be guaranteed that every further sub-partition would satisfy the above conditions. Therefore, in order to recursively partition A through several stages, we must insure that at least the semi-definiteness condition is satisfied by suitable multiplication. This makes the partitioning procedure very complex.

In order to obviate this difficulty, a recursive bi-section-transposition algorithm is suggested below based on the available results on g-inverses [1]. This algorithm does not depend upon the conditions (i) and (ii) since at any stage either a row or column partitioning of the matrix is carried out (and not both).

3. Bisection-Transposition Algorithm

a. Principle

Let the given matrix A_{r-1} (mxn) be partitioned as

$$A_{r-1} = [A_r | B_r]$$

where A_r(mxs) and B_r(mx(n-s))(s<n). Then it is easily proved that [1]

$$A^+_{r-1} = \begin{bmatrix} A^+_r - A^+_r B_r (C^+_r + D_r) \\ C^+_r + D_r \end{bmatrix} \tag{10}$$

where

$$C_r = (I - A_r A_r{}^+)\ B_r \tag{11}$$

$$D_r = (I - C_r C^+_r)\ Q^{-1}_r B^t_r (A^+_r)^t A^+_r (I - B_r C_r{}^+) \tag{12}$$

$$H_r = (I - C_r C_r{}^+)\ B^t_r (A_r{}^+)^t \tag{13}$$

$$P_r = H_r\ H^t_r \tag{14}$$

$$Q_r = (I + P_r) \tag{15}$$

Remarks

(i) When A_r is non-singular, we get $C_r = 0$ and $A^+_r = A^{-1}_r$ and $D_r = (I + B^t_r (A^+_r)^t A^+_r B_r)^{-1}\ B^t_r (A^+_r)^t A^+_r$. When B_r is a column vector, C_r is also a column vector and hence $C_r C^+_r = 1$ and hence $D_r = 0$.

Therefore, we get

$$A^+_{r-1} = \begin{bmatrix} A^+_r - A^+_r B_r C^+_r \\ \\ C^+_r \end{bmatrix} \qquad (16)$$

Equation (16) corresponds to the discrete Kalman filtering equations. (Kalman filtering is used extensively to estimate the internal state variable of a linear system based on noisy measurements of output variables.)

ii) In (12) and (13) certain matrices need not be conformable; this is taken care of by appending the required zero rows and columns to the smaller matrices [3].

b. Algorithm

Let A_0 be the given (mxm) matrix and let (pxp) be the order to which we finally desire to reduce this matrix. Then the number of partitions involved is $r = 2 \log_2 (\frac{m}{p})$; the number of transpositions required is (r-1). The algorithm uses any one of the standard procedures for computing A^+_i [4, 5].

In this algorithm $\rightarrow$ indicates the partitioning operation and $\leftarrow$ the use of (10) to obtain the pseudoinverse of the preceding larger matrix.

Step 1: Set $i = 1$; $A_0 \rightarrow [A_i|B_i]$

Step 2: $A^t_i \rightarrow [A_{i+1}|B_{i+1}]$

Step 3: Set $i = i+1$

Step 4: Is $i \leq r$; if yes go to Step 2; otherwise go go Step 5.

Step 5: Set $j = r$; compute A_j^+ .

Step 6: $(A_{j-1}^t)^+ \leftarrow (A_j | B_j)$.

Step 7: Set $j = j-1$

Step 8: Is $j = 0$? If yes, go to Step 9; otherwise go to Step 6.

Step 9: Result $= A_0^+$. Stop.

<u>Note</u>:

In partitioning, it is not required to carry out exact bisection of the matrix A_i; however, it is convenient to have exact bisection, and to have the order of A_0 be a power of 2.

4. Example

$$A_0 = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

A_0 is not positive semi-definite and $r(A_{11}) + r(A_{22}) \neq r(A)$.

If we take $A_1 = \begin{bmatrix} 11 \\ 11 \end{bmatrix}$; $A_{12} = \begin{bmatrix} 01 \\ 10 \end{bmatrix}$

$A_{21} = \begin{bmatrix} 01 \\ 10 \end{bmatrix}$; $A_{22} = \begin{bmatrix} 11 \\ 11 \end{bmatrix}$. Therefore, the classical four partitioning scheme will fail.

We now use the successive bisection-transposition algorithm; here $m = 4$, $p = 2$, $r = 2$.

Take

$$A_1 = \begin{bmatrix} 11 \\ 11 \\ 01 \\ 10 \end{bmatrix} ; \quad B_1 = \begin{bmatrix} 01 \\ 10 \\ 11 \\ 11 \end{bmatrix}$$

$$A_1^t = \begin{bmatrix} 1101 \\ 1110 \end{bmatrix} ; \quad A_2 = \begin{bmatrix} 11 \\ 11 \end{bmatrix} ; \quad B_2 = \begin{bmatrix} 01 \\ 10 \end{bmatrix}$$

$$A_2^+ = \frac{1}{4}\begin{bmatrix} 11 \\ 11 \end{bmatrix}$$

$$C_2 = (I - A_2 A_2^+)\, B_2 = \frac{1}{2}\begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$C_2^+ = \frac{1}{2}\begin{bmatrix} -1 & -1 \\ 1 & -1 \end{bmatrix}$$

$$H_2 = (I - C_2C_2^+)\ B_2^t(A_2^+)^t = \frac{1}{4}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$P_2 = H_2H_2{}^t = \frac{1}{8}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$D_2 = (I - C_2^+C_2)Q_2^{-1}\ B_2^t(A_2^t)A_2^+(I - B_2C_2{}^+) = \frac{1}{10}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

$$A_1^+ = \frac{1}{5}\begin{bmatrix} 1 & 1 & -2 & 3 \\ 1 & 1 & 3 & -2 \end{bmatrix}$$

Using A_1^+, A_1 and B_1 we get

$$C_1 = (I - A_1A_1^+)\ B_1 = \frac{1}{5}\begin{bmatrix} -4 & 1 \\ 1 & -4 \\ 3 & 3 \\ 3 & 3 \end{bmatrix}$$

$$C_1^+ = \frac{1}{3}\begin{bmatrix} -2 & 1 & 1 & 1 \\ 1 & -2 & 1 & 1 \end{bmatrix}$$

$$[I - C_1^+C_1] = 0;\ D_1 = 0$$

and

$$A_0^+ = \frac{1}{3}\begin{bmatrix} 1 & 1 & -2 & 1 \\ 1 & 1 & 1 & -2 \\ -2 & 1 & 1 & 1 \\ 1 & -2 & 1 & 1 \end{bmatrix}$$

5. Concluding Remarks

(i) The above algorithm is useful when one deals with matrices having certain special block features.

(ii) A fast algorithm for transposition is available in [6].

(iii) The numerical stability of the above procedure is not established.

It is well-known that the numerical algorithms for computing pseudoinverses lack numerical stability, and accordingly, error-free exact calculations are desirable [5]; see also [7].

(iv) The computational complexity of the above algorithm is to be established [8].

References

[1] R. M. Pringle and A. A. Rayner, "Generalized Inverses of Matrices", New York, Hafner Publishing Company, 1971.

[2] W. K. Pratt and F. Davarian, "Fast Computational Techniques for Pseudoinverse and Wiener Image Restoration", IEEE Trans. Comput., Vol. C-26, No. 6, pp. 571-580, June 1977.

[3] R. E. Cline, "Representations for the Generalized Inverse of a Partitioned Matrix", SIAM J. Appl. Math., Vol. 12, pp. 588-600, September 1964.

[4] A. Ben Israel and T.N.E. Greville, "Generalized Inverses Theory and Applications", New York, John Wiley, 1974.

[5] T. M. Rao, K. Subramanian and E. V. Krishnamurthy, "Residue Arithmetic Algorithms for Exact Computation of g-Inverses of Matrices", SIAM.J. Num. Analysis, Vol. 13, No. 2, pp. 155-171, April 1976.

[6] D. C. Van Voorhis, "Comments on a Computer Algorithm for Transposing Non-square Matrices", IEEE Trans. Comput. (Corresp.), Vol. C-26, No. 6, pp. 607-608, June 1977.

[7] S. L. Campbell, "On the Continuity of the Moore-Penrose and Drazin Generalized Inverses", Lin. Algebra and App., Vol. 18, No. 2, pp. 53-57, February 1977.

[8] A. V. Aho, J. E. Hopcroft and J. D. Ullman, "The Design and Analysis of Algorithms", Reading, Mass., Addison-Wesley, 1974.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFOSR-TR- 77-1252	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) RECURSIVE COMPUTQTION OF PSEUDOINVERSE FOR APPLICATIONS IN IMAGE RESTORATION AND FILTERING		5. TYPE OF REPORT & PERIOD COVERED Technical
		6. PERFORMING ORG. REPORT NUMBER TR-580
7. AUTHOR(s) E. V. Krishnamurthy		8. CONTRACT OR GRANT NUMBER(s) AFOSR-77-3271
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Science Ctr. Univ. of Maryland College Pk., MD 20742		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Math.& Info. Sciences, AFOSR/NM Bolling AFB Wash., DC 20332		12. REPORT DATE September 1977
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(If different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Filtering
Generalized inverses
Moore-Penrose inverse
Partitioning
Recursive computation
Restoration

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A bisection-tranposition algorithm is described for the recursive computation of the Moore-Penrose inverse of a large matrix.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)